



c. Problem 5-4. Write the answers here:

a.  $AR \leftarrow PC$

- 1
- 2
- 3
- 4

b.  $IR \leftarrow M[AR]$

- 1
- 2
- 3
- 4

c.  $M[AR] \leftarrow TR$

- 1
- 2
- 3
- 4

d.  $AC \leftarrow DR, DR \leftarrow AC$  (done simultaneously. Two or more RTL operations separated by commas means that they are done simultaneously. The effect of the operations here is to **swap** the data currently in  $AC$  and  $DR$ . Note to program swapping between variables in C/C++ 3 assignment statements are needed. Try it.) This example needs some operation in the adder and logic circuit. Write after [5] below what values should go into the operation select inputs of Table 4-8, page 118.

- 1
- 2
- 3
- 4
- 5

d. Problem 5-5.

a

b

c

Problem 2: For **each** of the parts below, assume that the computer is started with  $AC = F7BC$ ,  $E = 0$ ,  $PC = 6$ ,  $M[7] = FFFF$ ,  $M[8] = 0009$ , and  $M[9] = C099$ . For each case below of a different instruction in location 6 of memory, write the **(1) general** and **(2) specific** micro-operations that are performed in a sequence of groups where the micro-operations in each group are performed all in one clock step. Write each specific numeric value transferred in hexadecimal. Figure out the numeric values from the computer's starting state information given above.

Label each group with the name of its clock step, " $T_0$ ", " $T_1$ ", etc.

In other words, to do one case, first write every micro-operation that is performed during the first clock step. Write both the general micro-operation (in Register Transfer Language, from Table 5-6) and the specific data that is transferred. Label this group of one or more micro-operations with  $T_0$ . Then, go on to the micro-operations performed in clock step  $T_1$ , etc. until the work for the case instruction is done and the computer is about to enter the Fetch step again. Then, please stop and go on to the next case.

For the purpose of this homework, assume the given same starting conditions above for each and every one of the parts below. This will avoid your being sidetracked by interactions between possibly mistaken results of earlier instructions and later ones.

a.  $M[6] = 1007$ , that is, ADD 007.

b.  $M[6] = 5008$ , that is, BSA 008.

c.  $M[6] = D008$ , that is, BSA 008 I (indirect).

Problem 3: Continue the above problem with three more cases:

a.  $M[6] = E008$ , that is, ISZ 008 I (indirect).

b.  $M[6] = 7040$ , that is, CIL.

c.  $M[6] = 7008$ , that is, SNA.

Problem 4: Mano page 171, problem 5-21 (control gates for register  $PC$ ). For this problem, the control gates that you derive are to go in the “Control logic gates” box of Figure 5-6, page 137. The corresponding problem for register  $AC$  is solved on page 165. (No credit for just copying stuff from fig. 5-20!) To do this problem, learn to understand Table 5-6; then translate the relevant RTL specifications into gates.) **Put your neatly drawn answer on a separate sheet of paper, with your name on it!**

Next week’s HW will have problem 5-22..

Practice Problem B: (This problem will be assigned for credit on the next homework.) The organization and design of Mano’s basic computer have the property that *different kinds of instructions take different numbers of clock steps to complete*.

- a. Express in a table the number of clock steps used for each of the instructions in Table 5-2, page 133. Note that the micro-operation that makes the current clock step be the last for the current instruction is  $SC \leftarrow 0$ . Hint: Get the answers from page 159.
- b. The “Indirect” step can be skipped for direct addressing mode memory instructions. Therefore, the time for one clock step can be saved for such instructions. However, this improvement will require a redesign of some of the control functions. Explain how to change Table 5-2 to implement this improvement.